Submission instructions:

- If a problem asks for a numerical answer, you need only provide this answer. There is no need to show your work, unless you would like to.
- Please type up your solutions. We suggest using an online latex editor like www.overleaf.com, though this is not a requirement.
- Upload the PDF file for your homework to gradescope by midnight on Wednesday January 23.

1. **Casting an image into vector form.** A 10\times10 greyscale image is mapped to a \(d\)-dimensional vector, with one pixel per coordinate. What is \(d\)?

2. **The length of a vector.** The Euclidean (or \(L_2\)) length of a vector \(x \in \mathbb{R}^d\) is

\[
\|x\| = \sqrt{\sum_{i=1}^{d} x_i^2},
\]

where \(x_i\) is the \(i\)th coordinate of \(x\). This is the same as the Euclidean distance between \(x\) and the origin. What is the length of the vector which has a 1 in every coordinate? Your answer may be a function of \(d\).

3. **Accuracy of a random classifier.** A particular data set has 4 possible output labels, with the following frequencies:

<table>
<thead>
<tr>
<th>Label</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A)</td>
<td>50%</td>
</tr>
<tr>
<td>(B)</td>
<td>20%</td>
</tr>
<tr>
<td>(C)</td>
<td>20%</td>
</tr>
<tr>
<td>(D)</td>
<td>10%</td>
</tr>
</tbody>
</table>

(a) What is the error rate of a classifier that picks a label \((A,B,C,D)\) uniformly at random, i.e. each with probability 1/4, regardless of the input?

(b) An even simpler classifier is one that always returns the same label, regardless of the input.
   - For this particular dataset, in order to minimize error rate, what label should this type of classifier return?
   - What will its error rate be?

4. **Decision boundary of the nearest neighbor classifier.** In this problem,
   - The data space is \(\mathcal{X} = [0,2]^2\): each point has two coordinates, and they lie between 0 and 2.
- The labels are $\mathcal{Y} = \{1, 2, 3\}$.

The correct labels in different parts of $\mathcal{X}$ are as shown below.

(a) What is the correct label of point $(0.5, 0.5)$?

Now suppose you have a training set consisting of just two correctly labeled points, located at

$$(0.5, 0.5), (0.5, 1.5)$$

and that you are going to use a nearest neighbor (1-NN) classifier based on Euclidean distance (a.k.a. $\ell_2$ distance).

(b) What label will the nearest neighbor classifier assign to point $(1.5, 0.5)$?

(c) What label will the nearest neighbor classifier assign to point $(2, 2)$?

(d) Which label will this classifier never predict?

(e) Now suppose that when the classifier is used, the test points are uniformly distributed over the square $\mathcal{X}$. What is the error rate of your 1-NN classifier?

5. In the picture below, there are nine training points, each with label either square or star. These will be used to guess the label of a query point at $(3.5, 4.5)$, indicated by a circle.

Suppose Euclidean distance is used.
(a) How will the point be classified by 1-NN? The options are square, star, or ambiguous.
(b) By 3-NN?
(c) By 5-NN?

6. We decide to use 4-fold cross-validation to figure out the right value of $k$ to choose when running $k$-nearest neighbor on a data set of size 10,000. When checking a particular value of $k$, according to the standard cross-validation procedure, we will look at four different training sets. What is the size of each of these training sets?

7. An extremal type of cross-validation is $n$-fold cross-validation on a training set of size $n$. If we want to estimate the error of $k$-NN, this amounts to classifying each training point by running $k$-NN on the remaining $n-1$ points, and then looking at the fraction of mistakes made. It is commonly called leave-one-out cross-validation (LOOCV) or jackknife validation.

Consider the following simple data set of just four points:

```
+  +  -  +
```

What is the LOOCV error for 1-NN? For 3-NN?

8. Implementing $k$-NN. This problem will help familiarize you with Python if you don’t know it already and also give you some hands on experience with $k$-NN classification. I highly recommend using Jupyter notebooks for this problem and this course, though it is not required. You need not turn in your code. We only ask you to provide answers to the questions below, which will include experimental results from running your implementation.

We are going to look at the task of classifying images of digits using $k$-NN classification. Download the files hw0train.txt, hw0validate.txt and hw0test.txt from the class website. These files contain your training, validation and test data sets respectively.

For your benefit, we have already converted the images into vectors of pixel colors. The data files are in ASCII text format, and each line of the files contains a feature vector of size 784, followed by its label. The coordinates of the feature vector are separated by spaces.

(a) Implement a 3-NN classifier using Euclidean distance. Break ties by choosing the lowest digit. What is the error rate of your classifier when you use the provided training data (hw0train.txt) and evaluate on the validation data (hw0validate.txt)? This is called the validation error. [Hint: As a check for your code, the training error for a correctly implemented 3-NN classifier on this data should be approximately 0.04 – no need to match this exactly, it is just a rough guide.]

(b) For $k = 1, 5, 9$ and 15, build $k$-nearest neighbor classifiers from the training data. For each of these values of $k$, write down a table of training errors (error on the training data) and the validation errors (error on the validation data). Which of these classifiers performs the best on validation data? What is the test error of this classifier?

(b) Try your own distance metric! Implement a distance of your own design (other than Euclidean distance). Your distance need not be a formal metric (i.e. satisfying the mathematical requirements for a metric function), but you may find that non-metrics behave strangely in practice. Feel free to using validation error as a way to tune your choice. Report the best test error you achieve!